台科大
TAIWAN TECH

MASTER'S DEGREE IN INDUSTRIAL TECHNOLOGY ENGINEERING

Master's Thesis

SMART TOKEN FOR THE CAMPUS USING BLOCKCHAIN

Smart Campus Project

Author: Miguel Díaz, Javier Massó, Antoni Pérez-Villegas, Antoni Sagalés

Director: Shuo-Yan Chou

June 2017

National Taiwan University of Science and Technology

NTUST

Table of Contents

1. PREFACE

1.1. Origins

This project was born from the idea of creating a complete and functional Smart Campus on the National Taiwan University of Science and Technology (from now on NTUST).

The NTUST, one of the most excellence universities not only in Taipei City, but also all around Taiwan, has a large reputation in all types of engineering. Along the years, the university has been able to adapt to new technologies and new trends. In this new world wide environment of Internet of Things (IoT), the university wants to lead an interdepartmental project to create a SMART CAMPUS as functional and effective as possible.

According to this objective, between the writer and Professor Shuo-Yan Chou the idea of implementing a micro-economy system based in new technologies such as blockchain was thought as a revolutionary further step to develop in the project.

1.2. Motivation

Blockchain technology is barely introduced to the engineering society even though its whole potential is still a glance of the tip of the iceberg.

Nowadays, blockchain technology is gaining more and more power and social repercussion. Its new way of thinking allows developing multiple new applications. As an engineer, trying to understand blockchain and see and use its potential may be crucial for the future development of smart cities.

According to this, the writer wanted to go in deep in this knowledge and try to apply the blockchain technology to the Smart Campus Project of the NTUST University.

1.3. Previous knowledge

As we are in very early stages in this field of engineering there is not yet a very deep knowledge of this technology, in order to deal with this project, some background information is needed.

First of all, it is important to understand how blockchain works. Once achieved, knowing how to create a private blockchain and make it run with an own Token will be the point of the project.

2. INTRODUCTION

The National Taiwan University of Science and Technology is established as a world-class university of multifaceted excellence through international outreach and applied research.

In order to make the university more efficient, concerned, adapt it to the use of new technologies incoming and make it more innovative Professor Shuo-Yan Chou among other professors came to the idea of creating a Smart Campus. This project involves interdepartmental efforts, ideas and new technologies to reach the main goal.

Some of the projects concerning this **Smart Campus** are faced in developing apps to report maintenance problems; others want to use new technologies such as machine learning to facilitate the study of human flow; and other projects want to improve energy consumption by monitoring the usage of water dispensers.

This particular project sought the objective of creating a rewarding system and a micro-economy sharing society to be implemented in the smart campus. According to that, applying the newest technologies is the key point.

For the past few years, the technology of blockchain has been in upswing. This technology breaks with the centralized systems of nowadays to reach a decentralized network with peer-to-peer communication. Furthermore, blockchain enables users to create and use smart contracts that, among all of their characteristics, highlight the possibility of the creation of a token/cryptocurrency to be exchanged.

After a deep research in blockchain, a first brief introduction to the main characteristics of it is given to the reader. When the basic concepts of this revolutionary technology have been achieved, it is time to apply blockchain to the rewarding system project.

To do so, a private network based on the **Ethereum Platform** has been created. This network is sustained by a certain number of computers, working as nodes, that verify transactions happening across the network. Once the blockchain is created, the project goes through the creation of a token (named **NTUSToken**) that is given as a reward for making some good practices and is interchangeable by the students, giving them the sense of a micro-economy society based on the token. On this private blockchain, users can create accounts, transfer tokens and check their balances.

Finally, the last key point of the project is to make this system handy and attractive to users. As blockchain is not a trivial technology to understand, users have to be able to

register, log in to their accounts and share tokens without the necessity of knowing the technology behind it. For this purpose, a browser application connected to the private network has been created. With this application, users have an easy platform to join the micro-economy.

By now, the implementation of a micro-economy and a rewarding system in the NTUST has been achieved and can be implemented by following this thesis.

Some of the codes and files used are shown and detailed in the Annex.

3. BLOCKCHAIN

Many people may know blockchain as the technology held below the creation of Bitcoin in 2008. This particular cryptocurrency was the first one to enable online payments from one person to another without going through any financial institution or third-party.

Even though the first intention of its creators was to allow this peer to peer electronic cash system, the possibilities of blockchain go far beyond this matter. Nowadays blockchain is starting to be implemented all kinds of sectors and creating new applications.

3.1. Fundamentals

According to its primary origins, blockchain is commonly described as a distributed ledger where every update or movement on the chain is saved on all the components of the ledger.

This new technology can also be defined as a cryptographically secured database shared across thousands of computers. Blockchain breaks with centralized server networks to enable a decentralized system with plenty of applications.

As mentioned before, this technology is based on a decentralized chain. This chain is majorly formed by computers spread all around the world that are constantly sharing information between them and storing this information in blocks.

These computers sustaining the blockchain are known as nodes, and as long as one node is running, the entire blockchain will be running.

There exist two types of blockchain: **public**, where everyone can join and read and write data; and **private**, where only some trusted and known participants can access.

No matter if the blockchain is public or private, it is important to understand its two big components: **transactions** and **blocks**.

Transactions are the actions of sharing information and blocks are where all this information is stored. These blocks are essential for the blockchain because they also save a record of time when these transactions were added to the blockchain. Furthermore, when one block is formed it is added on top of the other blocks, which means that the chain only can be extended and previous records cannot be changed. This fact, related with the time stamp, makes the blockchain immutable. Moreover, as it is a distributed ledger, the blocks are not only formed in one computer but a copy of each block is updated in all the computers sharing the ledger.

Apart from transactions and blocks, it is also necessary to understand the meaning of keys. There exist two types of keys: **public keys** and **private keys**.

The first ones are the ones that are given to whom has created an account on the blockchain. This public key is the personal address of the new account and as its name suggests, it is public for everyone.

On the other hand, private keys are the ones that are only known by the owner of the account. These ones allow the owner to validate incoming and outgoing transactions.

Finally, the **hash** concept is something that has to be beard in mind. A hash is created every time a block is settled down. Each block has a different hash from the others. The hash is like the serial number to identify blocks along the network. Furthermore, this hash is what helps with the security of the blockchain as all the nodes are validating hashes from previous blocks ever since a transaction is trying to be executed.

Now that the main concepts of the blockchain have been explained, how a blockchain transactions works can be understood.

Two users of the blockchain, user A and user B want to exchange information between them. Using their public and private keys, the information requested to go from one peer to another is started to be validated by all the nodes of the chain. Once the transaction is validated, a block is formed; and so is the hash of this block. Then, it is stored on top of the chain. Bear in mind that this block is copied in every node of the blockchain. Finally, the transaction between A and B is settled down.

The scheme down below shows more accurately how transactions work:

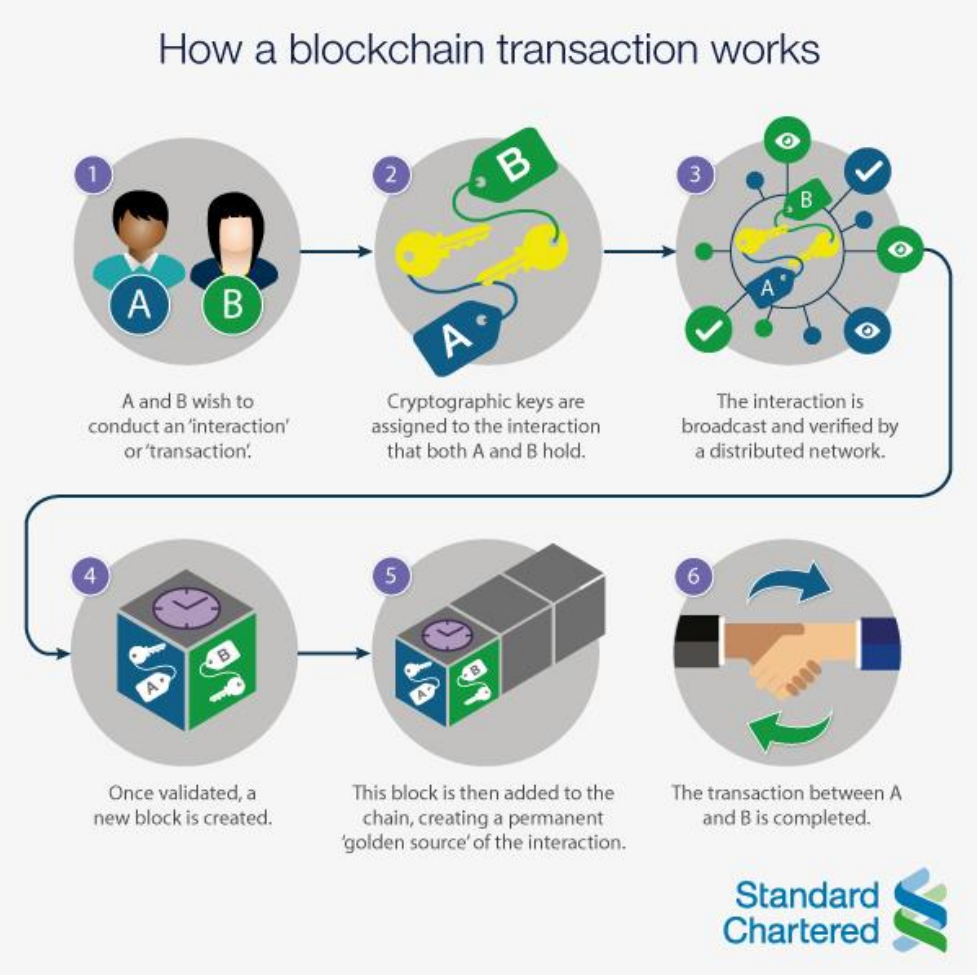

Figure 1: How blockchain transaction works. Source: Standard Chartered Bank.

However, at this stage is important to know that creating a block is not that simple. All the peers from the blockchain are constantly validating the information that is trying to be stored on the blocks. No matter if the blockchain is public or private, as this distributed system has no single owner; machines joining the network compete to validate transactions. This process is known as **mining**.

But mining requires some resources. As one may have thought, having a vast number of computers constantly validating transactions, ends up with a high energy consume which transforms into money consumption. That is the reason why miners (computers who do the mining) are paid a reward for each block they mine. This reward strictly depends on the blockchain they are mining.

3.2. Main characteristics

Once that the most important concepts of blockchain have been defined it is time to understand more in deep why this system is considered revolutionary by describing the main characteristics of blockchain.

3.2.1 Peer-to-peer

Peer-to-peer blockchain characteristic allows the network of computers to be connected between them without fixed single servers, but like a series of individual nodes. All those nodes are able to interact simultaneously as clients and servers in order to exchange information. This characteristic gives the blockchain security in front of failing points. This security field is explained in deep hereafter.

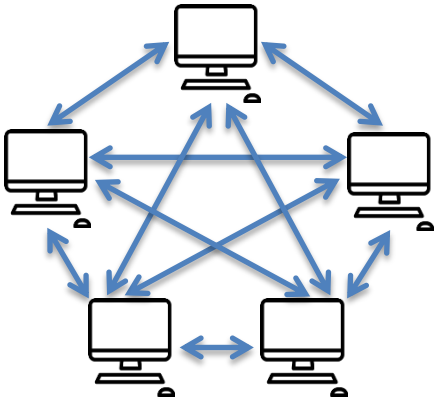

Figure 2: Peer-to-peer representation.

3.2.2 Decentralized and Trustworthy

For the past years all the platforms have been settled under a centralized structure where a mother-server stores all the information regarding different devices. However, blockchain breaks with this way of thinking and relies on a decentralized network where every node of the network is able to directly connect to others. Moreover, all nodes store the same exact information; in this case, the same blocks.

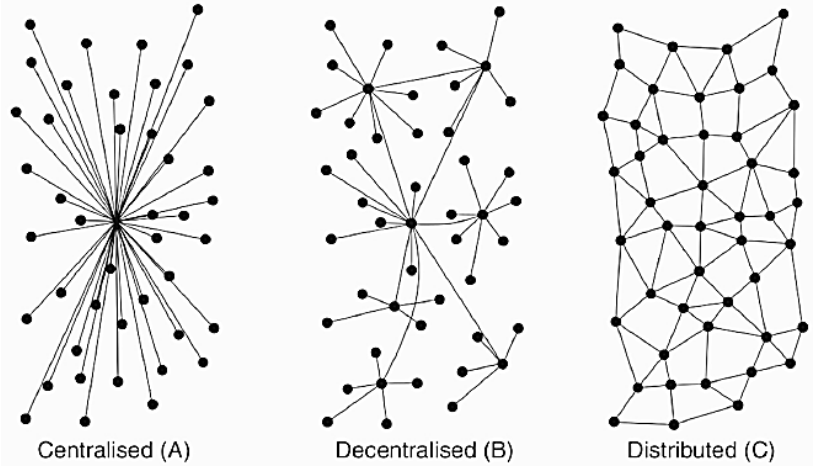

Figure 3: Networks types of distributions.

According to this characteristic, it can be concluded that blockchain is a trustworthy system as all the nodes of the network store the same information. Furthermore, new

blocks can only be added to the chain, and blocks that already belong to the chain cannot be erased or modified. Everything added to the blockchain will remain immutable.

3.2.3 Security

Regarding all the characteristics mentioned before, gathered together already help to describe the blockchain as a secure system to share any kind of information.

Additionally, what also makes blockchain a secure system is the way a transaction works. Whenever a transaction is trying to be done, this one has to be validated by the other nodes of the blockchain. Besides, the transaction needs of both the public and private key of each component of it to be executed.

Furthermore, the blockchain is formed by lots of nodes (as much as they want to join) sharing information. To shut down all the blockchain it would be necessary to shut down all of the nodes so, as long as one node is still running, the blockchain will still be perfectly functional.

3.3. Applications

This new concept of blockchain allows the creation of a vast range of different applications. Some of them are following described. As this project is based on Ethereum Project, the description of the following applications will be based in its possible implementations.

3.3.1 Smart Contracts

One of the most powerful characteristics for using the Ethereum Project is the creation of smart contracts.

Smart Contract is a term used to describe computer program code that is capable of facilitating, executing, and enforcing the negotiation or performance of an agreement (i.e. contract). The entire process is automated and can act as a complement, or substitute, for legal contracts. The terms of the smart contract are recorded in a computer language as a set of instructions.

This applications that run on either public or private blockchain, which do exactly as programmed without any possibility of downtime, censorship, fraud or third party interference. Smart contracts aim to provide security superior to traditional contract law and to reduce other transaction costs associated with contracting.

Lately, one of the most typical smart contracts applications has been the creation of cryptocurrencies.

3.3.2 Cryptocurrencies

As it has been cited above, blockchain is the technology behind most of the cryptocurrencies. That is why many new cryptocurrencies are emerging basing its technology on Bitcoin's.

Even though Bitcoin is the most known cryptocurrency that allows real payments nowadays, other cryptocurrencies are gaining power and market with its features and possible applications. For instance Ether, which is the second most powerful cryptocurrency, is becoming popular due to the Ethereum blockchain behind it, which grants creating simple smart contacts to be implemented. This project is based in Ethereum Blockchain.

3.3.3 IOT

Nowadays the **Internet of Things** (**IOT**) is creating new opportunities and providing a competitive advantage to many enterprises assembled in all different types of markets.

The new technologies permit the creation of more revolutionary devices which are required to be connected to internet. The combination of more devices and more connectivity ends up with the need of managing and storing more data.

All this data recorded has to be securely stored and transferred to the right place, time and format. At this point is where the blockchain concept appears as the optimum platform to interconnect IOT devices.

Blockchain will move from the current centralized ecosystem and let IOT devices to interact and share information and data between them. This will erase the big data centers currently used.

This decentralized approach would eliminate single points of failure, enabling the creation of a more resilient ecosystem for devices to run on. Besides, the cryptographic algorithms used by blockchain would make consumer data more private.

Also a fast and light payment method between Machine to Machine (M2M) payments is required for the good behavior of the IoT. We are approaching a future where machines would automatically pay for its electricity and buy its resources (for example a fridge will be able to buy milk if it detects that you have been run out of it). Blockchain can solve this problem by automatizing this process and eliminating the third parties and intermediates.

4. TOKENIZATION OF SMART CAMPUS

Some members of different departments of the National Taiwan University of Science and Technology have been working together to create a fully operative Smart Campus.

Concerning this Smart Campus, some of the projects are developing systems to make life at the university more environmentally friendly and better such as the **Crowdsourcing Maintenance System** and **Energy Management System**; others want to monitor human and traffic flow by developing **Transportation Apps**; and also platforms for **Sharing Space Usage** are being sought.

All these projects have the objective and the need to be used and integrated by university users. Without users, the Smart Campus is not able to grow and evolve.

Since the beginning of the history, human beings have always been more attracted to participate in doing actions when something is given in return. Humans want to be rewarded whenever they contribute to the well behavior of the society. Based on this idea, the Smart Campus project wanted to have a rewarding system which enhances students to use the platforms created and to contribute more deeply in the project.

Here is where this particular project comes to a reality. The "**Smart Token for the Campus Using Blockchain**" project wants to be part of the Smart Campus by creating a decentralized rewarding system capable of awarding and retrieving tokens to the university students depending on their behavior.

This project wants to take advantage of the brotherhood projects of the Campus to be implemented. It works with synergy with the other projects relying on their platforms to be included on the rewarding system.

As the university is a public institution, no centralized database or administration power is wanted. The main objective is then to develop a system that can run itself with no human interaction. Hence, the idea of the blockchain technology explained before materializes itself as the answer to the prayers. This technology behind the rewarding system is accurately explained on the following chapters.

For the rewarding system, a crypto-token has been created using the smart contract features of Ethereum's platform. For now on, this token is known as **NTUSToken**. This new token is the one that is given to the users, or retrieved, depending on their actions. Moreover, when some tokens are earned, the system also allows transacting tokens between parties to create the feeling of an active micro-economy inside the university.

As mentioned, this token wants to take part in all the projects related with the Smart Campus but, by now, some of these are still on-going. That is why to start implementing this token other activities and resources have been taken into account. Nonetheless, the system allows adapting new activities whenever the other projects are fully operative.

The activities that, for the moment, permit earning or losing NTUSTokens are those university activities that involve the use of the personal Easy Card. With the scan of this card on the readers, the system is able to identify which action is being done and gives or retrieves tokens accordingly. By now, some possible implementations for the usage of the token are:

- **Library**
  When a student goes to the library, he/she has to scan his Easy Card to enter. This action is qualified as a good action that rewards the student with 10 NTUSTokens
- **Swimming pool**
  The same happens with the swimming pool. Whenever used the system rewards the student with 10 NTUSTokens
- **Usage of parking lot**
  Contrarily to the two activities mentioned above, abusing of the usage of parking lot retrieves 5 NTUSTokens from the user.

Both going to the library and to the swimming pool are considered good actions that are rewarded because the university, in a backend way, earns money or revalorizes its inversion with them.

On the one hand, rewarding people going to the library make students more academically involved and gain excellence, which will help the university with its excellence among other universities according to better qualified students graduated.

On the other hand, students using the swimming pool enhance a sportive life style which has a tied correlation on reducing healthy costs.

However, the abuse of usage of the parking lot is seen as an activity that concludes in retribution of tokens for the users. The reason of that “bad action” qualifier is because of an environment factor. Cars pollute more than public transport alternatives, hence, and abuse of car ends up with damaging the environment. This way, users may start thinking about taking into account other alternatives to go to the university instead of cars.

As mentioned above, once the users earn tokens, they are able to interact and transact them to other users. These transactions may be executed in reward when a student helps another one studying for an exam or when sharing university tools such as calculators or computers.

By now, users of the Smart Campus are able to interact with the system, be rewarded or retrieved NTUSTokens and participate on the micro-crypto-economy created.

A further step on this matter is to set some basis to let users exchange their NTUSTokens with something more real. For instance, discounts on paying conferences or discounts on the university tuition.

5. CREATING A PRIVATE BLOCKCHAIN

As the aim of the project is to create a rewarding system within the Smart Campus, a private network is required in order to avoid paying the fees to the miners. The so-called private network is run by the nodes whereas the end-users carry a wallet account.

The above-mentioned private blockchain is built on top of the Ethereum open source platform, which allows to create smart contracts and to transact information separately to the main public network. Find the code lines for this purpose in the ***Annex 9.2***.

5.1. Genesis

The genesis block is the first block of the blockchain and, therefore, is the only one that has no predecessor. Defining the Genesis block is the key to determine the block creation velocity, as well as the fees that the mining power will get in return of its computational power. These features are determined by the so-called difficulty of the block chain.

In order to add a node to the private network it has to initialize this private network with the same genesis block of the other users.

Another important feature that the genesis block must contain is the **chainId**, which determines the port where the other nodes will be connected.

The genesis block used in this project can be found in the ***Annex 9.1*** with brief explanations of the parameters used in this project.

5.2. Nodes

It is fundamental to distinguish in between full nodes and lightweight nodes of a specific blockchain. Full nodes form the backbone of the network, they storage and run the whole blockchain ledger.

Their main function of the **full nodes** is to ensure that all the consensus rules are achieved. The basic rules are the following:

- Blocks may only create a certain number of tokens.
- Transactions must have correct signatures for the tokens to be spent.
- Transactions and blocks must be in the correct data format.
- Within a single blockchain, a transaction output cannot be double-spent, otherwise someone could unbalance the limited number of tokens spread and hence, would be defrauding.

If two or more full nodes have a different consensus rules are actually using two different networks/currencies. Changing the consensus rules requires a hard fork, for further information check the reference.

On the other hand, **lightweight nodes** are those clients that have access to the network, run the network but do not storage nor guarantee that the consensus are reached. Lightweight nodes rely on third parties (full nodes) that guarantee the robustness of the system. In other words, lightweight nodes are the wallets, where the users can have different accounts.

5.3. Accounts

There are two different types of accounts in Ethereum:

- **External accounts** are controlled by public-private key pairs. The private key is personal, as far as the network is concerned; anyone with your private key is you. Even if your wallet gets destroyed, as long as you keep the private key you can restore your digital assets.
- **Contract accounts** which are controlled by the code stored together with the account.

5.4. Connecting peers

Once the nodes that run and secure the blockchain are connected to the same network, the next step is to connect them in order that they become peers. The administrator has to add the peers using the peer identifier, the so-called enode.

In order to automatize this process of adding peers, a Static.nodes.json is created with all the nodes identifiers (enodes) of the nodes of the network. To see the Static.nodes.json, see the ***Annex 9.3***.

5.5. Transactions

In any transaction of information, encryption leads to trust. Asymmetric cryptography is a method of sending secure messages back and forth over a network, when the communication channel cannot be trusted.

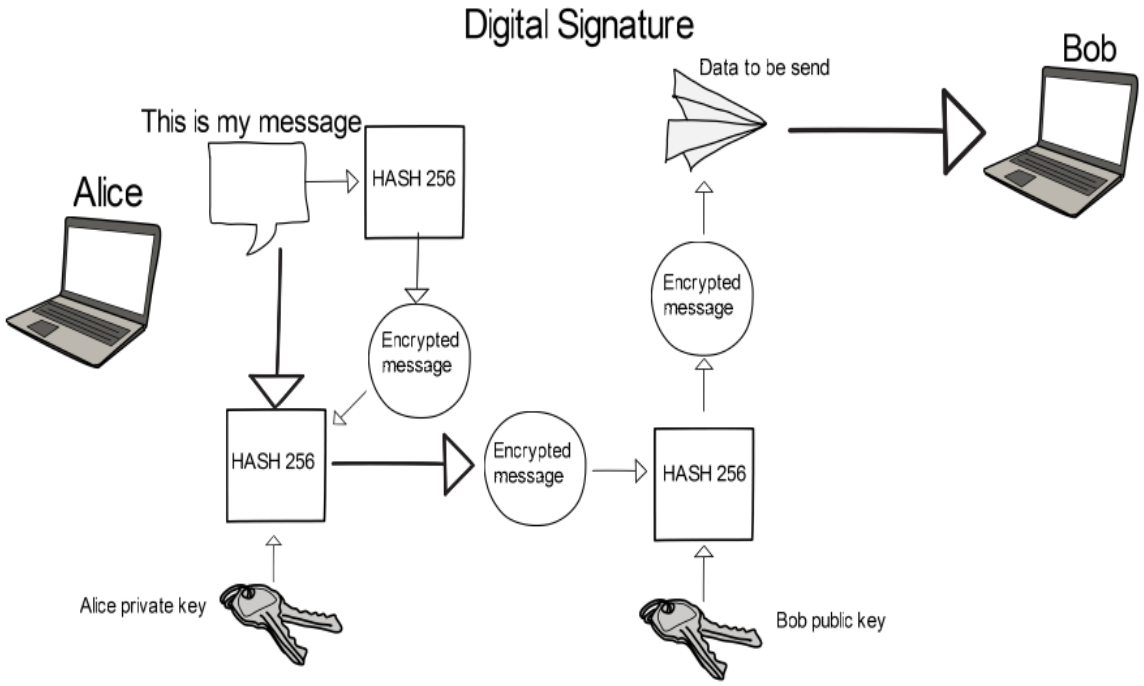

Figure 4: Encryption of transactions.

To describe how the secure and signed messaging works, the image above explains graphically the technology behind a transaction on the blockchain. Alice wants to send a transaction to Bob. To do so, she first encrypts the message using her private key, afterwards, the encrypted message is encrypted again by Alice using Bob's public key. When the data arrives to Bob, he decrypts the first layer with his private key and then uses Alice public key to verify that she, indeed, sent the message (because otherwise the message could have been sent by another user claiming that was Alice).

5.6. Data Base

One of the benefits of using blockchain is the possibility to store the users database on it. So the reader may deduce that the blockchain is not only for keeping an immutable track of the transactions but also permits the storage of data in it.

Why is keeping the users database stored in the blockchain a good idea? This question can be answered using some of the key points of blockchain technology:

- Keeping the database in the blockchain provides **decentralization**, meaning that while there are different nodes running the network if any of them has a problem (e.g. the electricity of the building is shut down) the other ones keep the information secure, regarding that fact, one blockchain can be run with several different nodes distributed all over the world. This decentralization provides security not only in case of failure but also in case of an attempt of corrupting the

blockchain, if someone tries to change the information stored in one node as long as the other nodes do not validate this modification in the blockchain it will not be submitted.

- Another benefit would be the **immutability** of the data stored. So far some data is stored in the blockchain it will be impossible to change or alter it. The blockchain is an immutable track of blocks storing information; this fact prevents future corruption or data alteration for malicious intentions.
- The **transparency** that provides the network is also a key point, each user of this network have access to the information stored in it so everyone can verify the correct performance of the network. Anyone can build some applications on top of the private network using this information that is 100% reliable.
- **Reliability** is also a very important aspect, if you want to provide trust in your system. Using the Ethereum platform there is no need to trust the other parties for verifying the information and the transactions, it provides 100% reliability and trust.

6. SMART TOKEN

A peer-to-peer private blockchain network has been successfully created. Now, it is time to introduce the **NTUSToken**.

The NTUSToken is the digital asset of the rewarding system that is built on the Smart Campus Project. The main function of creating a rewarding system is to enhance the life quality of the Smart Campus users as well as to improve the way they interact within the members of the community.

To make things clearer and as mentioned before, the NTUSToken will be delivered automatically (with the execution of Smart Contract) to these users that add up value to the community. For instance, the Smart campus users that study (by means of going to the library), that are healthy (by means of going to the swimming pool) will get NTUSTokens in return.

A lot of other applications of the Smart Campus can be built on top of the NTUSToken, for instance applications to reward saving energy consumption or rewarding a proper use of water dispensers.

6.1. Smart Contract to create own currency

DApps are fully decentralized applications by default, and so is the blockchain technology. The smart token that is being sought, then, has to be as decentralized as possible. Although it may seem easy to decentralize a system, it is not.

After some efforts to make the NTUSToken and the system decentralized, the best idea came to reality. This idea relies on the creation and destruction of tokens every time good or bad actions are registered to the blockchain.

At an early stage, the easiest solution to the smart contract is to create an administration account, owned by the university, who earns all the NTUSTokens created when the smart contract is deployed. Then, this administration account is in charge of rewarding and retrieving tokens to the users of the Smart Campus. The problem appears whenever the administration account is created because then, the system is not decentralized as the power remains in it.

Enabling the creation and destruction of tokens, the system dispense with the administration account. The Smart Campus becomes more decentralized because tokens are not owned in advanced by an institution but they are created. Those NTUSTokens are created every time a good action is done by a user; and are given as a reward to it. These

tokens rewarded are also added to the total supply of NTUSTokens. On the other way around, NTUSTokens are destroyed from the account of the user who does a bad action; and so are destroyed from the total supply. Moreover, with this system the Smart Campus is capable to control de inflation of the token however some further upgrades may be discussed according to this matter.

Under these main concepts and requirements, the basic functionalities of the Smart Contract deployed in the private blockchain to create NTUSTokens are the following:

- **Add User**

  This function is the first one that is called when someone creates an account. It initializes al the variables at 0 so that the user can start taking part of the Smart Campus. By variables it is meant the NTUSTokens amount and Penalty Points. This last will be introduced hereafter.
- **Mint Token**

  This function is vital for the surviving of the Smart Campus. Actually, the project relies on the mining of NTUSTokens whenever a good action is added to the blockchain.

  When the function is called, it mines tokens that are added to the user wallet who makes the action. This tokens mined are also included to the total supply of NTUSTokens.
- **Destroy Token**

  This function is executed when a bad action is added to the blockchain. Depending on the action made, the amount of tokens destroyed can be modified. Those tokens destroyed are also erased from the total Supply of NTUSTokens.
- **Transfer**

  This function allows exchanging tokens among the members of the community. For this purpose, it is only required to know the public key (user ID) and the private key (password) of the sender. Transactions can only be executed under some conditions. When the account of the sender is freeze, transactions will not execute.
- **Freeze account**

  When a continuous behavior of doing bad actions along the Smart Campus is adopted by some users, the freeze account is executed. An account is freeze when its Penalty Points counter exceeds a stipulated maxim of 25 **Penalty Points**. These points increase and decrease at the same time the NTUSTokes do.

  For good actions, the user is retrieved with 2 Penalty Points; which will, consequently reduce its **Penalty Counter**. For bad actions, the user receives 5

Penalty Points; increasing the counter. Notice that Penalty Points cannot be negative and are always greater than 0.

Accounts can also be frozen when the user is not a Smart Campus member anymore (it may be the case of exchange students).

- **Smart Users Payments**

  This function is based on the first two functions mentioned above. It automatically uses the previous functions in order to reward or retrieve NTUSTokens. The inputs of this function are the account of the user that has used its easy card and the easy card reader ID.

  The easy card reader ID determines the origin in which the user has interacted. The swimming pool has a different card reader ID than the library and both of them have a different ID than the parking lot. This ID is what enables the rewarding and retrieving different amounts of tokens as it can be changed.

The Smart Contract deployed in this project can be found in the ***Annex 9.4*** with comments and explanations of the code.

6.2. DAap

At this stage of the project, the objective of creating a private blockchain at the NTUST Campus has already been achieved. However, as explained before, this project is not just about that, but also about letting the students participate on a micro-economy and be capable to earn and share tokens among them.

Since now, all the procedures do not look friendly to a normal user; that is to say to a student. Blockchain is a new and complex concept that does not have to be understood by all the university members, but it has to be used among the students. It is here where the second stage of the project comes to a reality.

To help in this matter, a DAap has been created. This interface is a localhost connected to the blockchain through the Web3. This particular DAap is connected to the private blockchain and allows users to interact both with the network and among them.

This interface will be referred as the **NTUST Wallet**.

The interface looks as follows:

Figure 5: DAap. NTUST Wallet home page

Apart from the basic information that the wallet provides, such as contact addresses, introductory videos and related websites (***Annex 9.5***), this one helps the students in the following matters:

- **Create an account**

  The user introduces the easycard/student number and creates a password to enter his account.

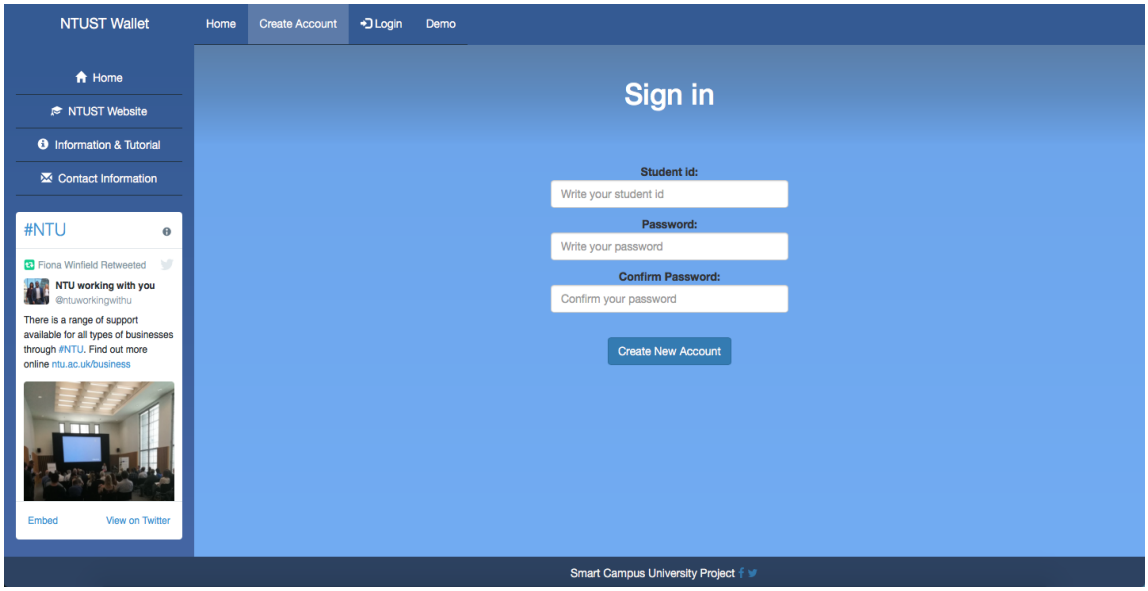

Figure 6: DAap. Sign in.

Once the account is created, this user receives a public key that enables him/her to receive and transfer NTUSTokens.

- **Log in**

To log in a personal account the student number ID is required. As blockchain claims to be a shared ledger that allows transparency, everybody will be able to check the balance of others but just the owner will be able to do transactions with his account.

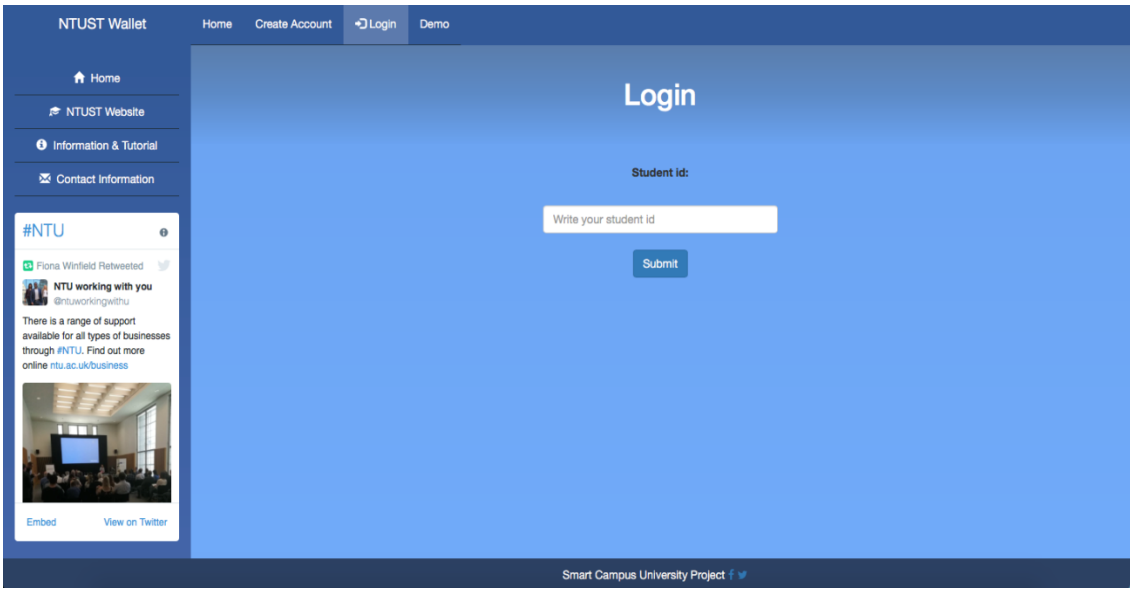

Figure 7: DAap. Log in.

When someone enters to the account it appears its Address, the balance of NTUSTokens, the amount of Penalty Points and an option to send tokens to other users. To do so, as explained above, the user only has to enter the student number of the receiver, the amount of tokens wanted to send and the personal password.

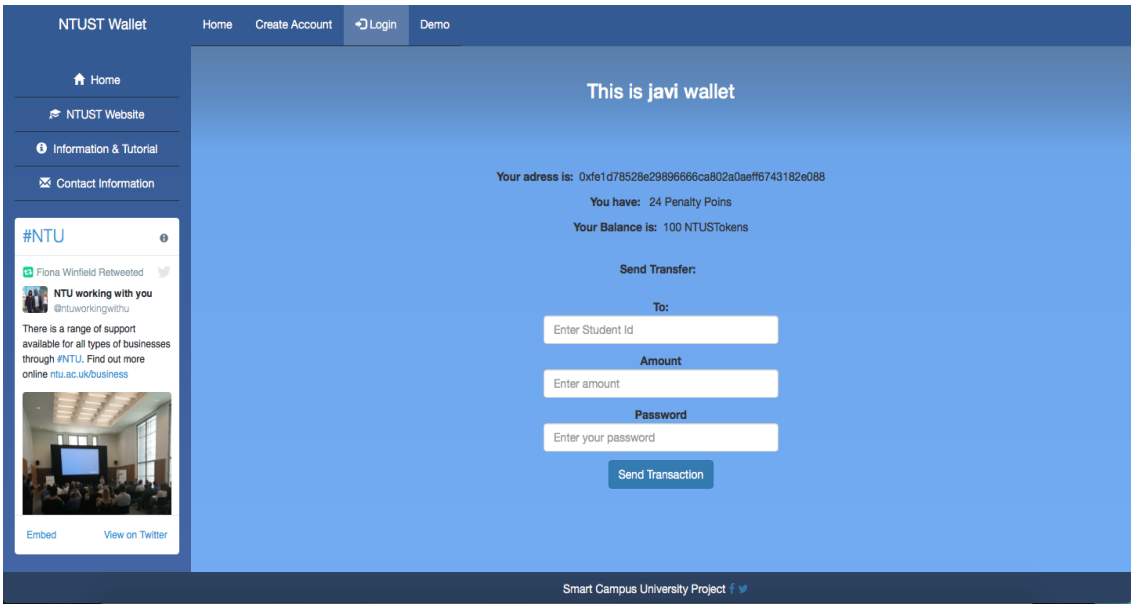

Figure 8: DAap. User Wallet.

6.3. File and image upload

Right now, the blockchain is already created and a DAap has been developed to ease the interaction of users. But blockchain has more potential in terms of technology. For instance, images and files can be uploaded to the network.

This characteristic of uploading files to a blockchain gives the Smart Campus project the capability of having immutable records; these records cannot be changed, manipulated nor altered. It also provides authentication of the author.

When uploading a file, this file is coded in base64 string, and then this data is recorded in the blockchain. When downloading a file, this one is downloaded in the base64 string which later on, in the DAap, it is decoded to the original file.

The amount of the gas limit per block had to be incremented, this gas limit is incremented in the genesis files, for test purposes it is increased to 5 quintillions of gas, though only a couple of millions of gas where used. Bigger files required more gas, a .txt file can have some millions of gas, but a .pdf file can consume as much as quadrillions of gas.

As for today's connectivity problems, large strings of data disconnect web3 application from the blockchain, so there's a limit size of a couple of tens of kilobytes to upload from the DAap. It is possible to upload bigger files directly to the blockchain, for example with the Ethereum official Wallet, a .pdf file of 550kb consumes more than 5 quintillions of gas.

The authentication part follows the same core system as a transfer function.

In order to call this upload function the user must unlock its own account, then a mapping of the address that is calling the function is done with an integer number, at the same time another mapping with an integer number is done with the data that is been uploaded. This integer number in both mapping represents the position of this list.

To verify the author is as simple as just looking for the same position in both lists. For example, if User A uploads a file its address will be automatically added to a list that occupies the same position of the data that is uploading in a different list, they occupied the position 7.

When retrieving this data, we search for the 7^{th} position of the lists. On the first list, Author A will be reached; and on the second list the data that User A uploaded will be reached.

The smart contract implemented in order to upload files and images to the blockchain can be found in the ***Annex 9.6***.

7. CONCLUSIONS

As the reader may have seen in this thesis the aim of this project is not only to start implementing blockchain technology in National Taiwan University of Science and Technology but also creating a platform that enhances the students to interact and start using all the different implementations of the Smart Campus Project. The creation of a digital Token based in blockchain technology is the best way, from the writer's point of view, for achieving that purpose.

Beyond the explanation of the functionalities of the NTUSToken and its wallet, one of the main purposes of the thesis is also create awareness of the blockchain technology among all the students and staff of the university. By introducing this new technology is intended to create an active environment of enthusiasts so that they can be able to start building new applications on top of the university's blockchain and keep promoting new innovative projects.

It is expected that the students will rapidly adopt both the use of the new wallet and the new micro-economy implemented behind the NTUSToken and soon they would use it as another tool provided by the university. This project pretends enhancing the use of university's facilities, encourage students to be part of more inner activities of the campus and reward the good actions and behaviours of them.

This project details all the steps from building a private blockchain network based on Ethereum's technology to the deployment of the Smart Contracts that create a digital asset and the implementation of a user-friendly interface to interact with it. A decentralized data base stored on the blockchain provides more security and trust to the system, therefore, it has been implanted and detailed in this project.

In conclusion, this project sets the basis of an active and promising blockchain atmosphere inside the university and promotes the university towards new innovative areas where, if intended, can internationally highlight.

8. FURTHER STEPS

The next steps of this implementation would include the total integration of the easyCard readers of the university with the private network, some promotional conferences and meetings in order to get this new tool to the students, the final setting and implementation of the different nodes of the network in the university and a trial period of implementation for receiving all the possible feedback in order to improve the system.

At this moment, the implantation of systems based on blockchain is scarce. The possibilities offered by this new technology are very large. There will be revolutionary effects in a huge range of different fields as varied as finance, contracts, law, voting systems business management and intellectual and industrial property. This is due to the high security, transparency and trust can be provided by this new thrilling technology.

It is safe to say that in the upcoming years blockchain technology will take part in more and more aspects of people's life so, it is in everyone's hands to start learning about it and start improving in some fields of daily life. This thesis tries to achieve this pursuit and increase the awareness between the future technologic profiles of the society which are the university students.

9. ANNEX

9.1. Genesis Block

As explained along the project, the Genesis block is the only one that has no predecessor. It is needed to create it and save it in a specific folder from which it will be called. The document is saved on a *.json* format.

The code used for the purpose of this project is the following:

```
{
   "config": {
      "chainId": 15,
      "homesteadBlock": 0,
      "eip155Block": 0,
      "eip158Block": 0
   },
   "difficulty": "200",
   "gasLimit": "2100000",
   "alloc": { }
}
```

9.2. Private Blockchain Code Lines

It has been talked about the genesis block and the blockchain main features. To help the programmer readers, some line codes to run the private blockchain are described down below.

***Note**: This whole project is based on the Ethereum platform. A new private blockchain is created on top of the Ethereum blockchain, so all the commands and clients used to achieve this purpose are those from the Ethereum Project Platform.*

In order to implement the blockchain, the easiest way is to use the Command Prompt in the computer. From here is where the Ethereum client **Geth** (Go Ethereum) is accessed. This particular client is public and can be downloaded from the ethereum.github website.

Once reached the folder where the *genesis.json* file has been saved (*MyDirectory*), it is time to initialize the genesis. To do so, the following code is inserted:

geth --datadir MyDirectory init genesis.json

Once the genesis file is initialized and the first node of blockchain is created, to enter the blockchain and see its potential, the geth console has to be run. The next code line helps on this matter:

geth --datadir MyDirectory --networkid## --nodiscover console

It is reminded that a private blockchain is being sought (even though this one runs on Ethereum public blockchain). That is the reason why the flags *networkid##* and *nodiscover* are executed. The first one will enable to connect to a particular blockchain; the second one is part of the security of the blockchain as it only allows computers with the same genesis to access it.

This whole initialization procedure is done at the same time by different computers sharing the same genesis file. Then, all the nodes will be created in the blockchain with its particular ***enode*** identification but, so far, they will not be connected. To enable these nodes to connect between them and create a peer-to-peer network, one previous step has to be done before running the console next time. Another *.json* file has to be created and saved in the same directory of the *genesis.json* file. This *static-node.json* file (***Annex 9.3***) enables different computers to be perfectly connected between them along the blockchain.

Once the geth console is running and the nodes have become peers, some further steps can be done.

An important point to do at this stage is to create and unlock accounts to use them. The following code lines help in this matter:

personal.newAccount()

The only thing required is a personal and unique **password.** In order to do transactions with an account, it is required to unlock it first:

personal.unlockAccount("AccountName"*)*

With the account unlocked it is time to start mining to create the synchronism of the blocks As a compensation for this mining process, the accounts earn ethers. It is reminded that these ethers are not economically worth as this blockchain works on a private test-net of Ethereum. To start and stop mining, these following lines may be used:

miner.start(1)

miner.stop()

The mining part is essential for letting the blockchain run. All transactions and actions made in the blockchain have to be constantly validated by the nodes and this is made while the mining is running. If there is no mining, the blockchain does not upload itself.

9.3. Static Nodes

The static nodes it is a *.json* file that once it is saved in the node it can be easily run with the geth. This *.json* file, called static-nodes.json, contains the enodes of all the nodes that will be peers. In this manner, every time the console is executed, all the peers will be added automatically.

The *static-nodes.json* might look like this:

```
[
"enode://27f5ea579b7e9956f92797a2ae672731594ccd6e7fe8c0b9eaf21b4051274325484436a68
77c992c66398324e044eeb725f1f24482e96e5374fa16c356f9e6cd@140.118.138.242:30303",

"enode://aee699ce020ffc266d3325950eb0b9454fb839abd87712324e8f2917e68fe94c2983dc165
cac46aa2c12abe18d38eccd1c54a7ad1142c709c347c6d228536e1b@140.118.134.240:30303",

"enode://71f8e43117d23a40c8241ed49993c3c0a4a900ced15458aa6426c923fbcebea8e50cf7595
80f949548de0d4e51519422ddb6d4fe7709186d3e4284b809f01d67@140.118.144.46:30303",

"enode://efd5e49b6ffae28d56bc995013510aad47a6cae3c0aaa346e452aa95157e8fd930a478b47
95f53f54b21ffae94f29efcd33ad8877f007181bed8587550e30602@140.118.137.170:30303"
]
```

It is important to bear in mind that the numbers that follow the “@” on every enode are the correspondent IP Addresses of every computer that is going to be a node. This numbers, hence, have to be changed accordingly.

9.4. Smart Contract: NTUSToken

```
pragma solidity 0.4.11;
contract owned {
   address public owner;

   function owned() {
      owner = msg.sender;
   }

   modifier onlyOwner {
      if (msg.sender != owner) throw;
      _;
   }
}
```

```
contract tokenRecipient { function receiveApproval(address _from, uint256 _value, address
_token, bytes _extraData); }

contract token {
    /* Public variables of the token */
    string public standard = 'Token 0.1';
    string public name;
    string public symbol;
    uint8 public decimals;
    uint256 public totalSupply;

    /* This creates an array with all balances */
    mapping (address => uint256) public balanceOf;
    mapping (address => mapping (address => uint256)) public allowance;

    /* This generates a public event on the blockchain that will notify clients */
    event Transfer(address indexed from, address indexed to, uint256 value);

    /* Initializes contract with initial supply tokens to the creator of the contract */
    function token(
        uint256 initialSupply,
        string tokenName,
        uint8 decimalUnits,
        string tokenSymbol
        ) {
        balanceOf[msg.sender] = initialSupply;              // Give the creator all initial tokens (by default
                                                                                    it has to be greater that 0)
        totalSupply = initialSupply;                        // Update total supply
        name = tokenName;                                   // Set the name for display purposes
        symbol = tokenSymbol;                               // Set the symbol for display purposes
        decimals = decimalUnits;                            // Amount of decimals for display purposes
    }

    /* Send coins */
    function transfer(address _to, uint256 _value) {
        if (balanceOf[msg.sender] < _value) throw;          // Check if the sender has enough
        if (balanceOf[_to] + _value < balanceOf[_to]) throw; // Check for overflows
        balanceOf[msg.sender] -= _value;                    // Subtract from the sender
        balanceOf[_to] += _value;                           // Add the same to the recipient
        Transfer(msg.sender, _to, _value);                  // Notify anyone listening that this transfer took
place
    }

    /* This unnamed function is called whenever someone tries to send ether to it */
    function () {
        throw;     // Prevents accidental sending of ether
    }
}

//////////////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////////////
```

```
//*************************************************************************************

contract MyAdvancedToken is owned, token {

    mapping (address => bool) public frozenAccount;
    mapping (uint => address) public findUserStep1;
    mapping (address => string) public findUserStep2;
    mapping (uint => string) public findUserEC;
    mapping (address => uint) public penaltyCount;

    uint public count;
    uint public countEC;

    /* This generates a public event on the blockchain that will notify clients */
    event FrozenFunds(address target, bool frozen);

    /* Initializes contract with initial supply tokens to the creator of the contract */
    function MyAdvancedToken(
        uint256 initialSupply,
        string tokenName,
        uint8 decimalUnits,
        string tokenSymbol
    ) token (initialSupply, tokenName, decimalUnits, tokenSymbol) {}

    /* Send coins */
    function transfer(address _to, uint256 _value) {
        if (balanceOf[msg.sender] < _value) throw;          // Check if the sender has enough
        if (balanceOf[_to] + _value < balanceOf[_to]) throw; // Check for overflows
        if (frozenAccount[msg.sender]) throw;               // Check if frozen
        balanceOf[msg.sender] -= _value;                    // Subtract from the sender
        balanceOf[_to] += _value;                           // Add the same to the recipient
        Transfer(msg.sender, _to, _value);                  // Notify anyone listening that this transfer took
place
    }

    /* A contract attempts to get the coins */
    function transferFrom(address _from, address _to, uint256 _value) returns (bool success) {
        if (frozenAccount[_from]) throw;                    // Check if frozen
        if (balanceOf[_from] < _value) throw;               // Check if the sender has enough
        if (balanceOf[_to] + _value < balanceOf[_to]) throw;  // Check for overflows
        if (_value > allowance[_from][msg.sender]) throw;   // Check allowance
        balanceOf[_from] -= _value;                         // Subtract from the sender
        balanceOf[_to] += _value;                           // Add the same to the recipient
        allowance[_from][msg.sender] -= _value;
        Transfer(_from, _to, _value);
        return true;
    }

    function smartUsersPayments(address _user, uint256 _origin) payable onlyOwner returns (bool
success) {
        if (_user == 0x0) throw;
```

```
if (balanceOf[msg.sender] < 10) throw;
if (balanceOf[_user] + 10 < balanceOf[_user]) throw;

uint mintedAmount = 10;
uint aux =0;
uint positivePoints = 2;
uint negativePoints = 5;
uint destroyAmount = 5;
if((_origin>0) && (_origin<= 100)){

   //the minning of this token are the good actions of the people
   balanceOf[_user] += mintedAmount; // user gets 10 token for good action
   totalSupply += mintedAmount;  //total supply increases in 10
   Transfer(0, this, mintedAmount);
   Transfer(this, _user, mintedAmount);

   aux = penaltyCount[_user]; //get the penalty count from the user
   if (penaltyCount[_user] == 1){
      penaltyCount[_user] = 0; //update the penalty number to 0
   }
   if (penaltyCount[_user] > positivePoints){
      penaltyCount[_user] = aux -positivePoints; //update the penalty number -2
   }
   // unfreeze account if point go lower than 25
   if (penaltyCount[_user] < 25){
      frozenAccount[_user] = false;
      FrozenFunds(_user, false);
   }
   return true;
}
else{
   //to prevent inflation everey bad action will destroy 5 coins from the total supply
   //this tokens are destroyed from the user account
   if(balanceOf[_user] >= destroyAmount){
      balanceOf[_user] -= destroyAmount; // user destroys 5 token for bad action
      totalSupply -= destroyAmount;  //total supply decreases in 5
      //Transfer(0, this, mintedAmount);
      //Transfer(this, _user, mintedAmount);
   } else {
      if((balanceOf[_user] > 0) && (balanceOf[_user] < destroyAmount)){
         uint aux2 = balanceOf[_user];
         balanceOf[_user]-= aux2; //balance of the user gets to 0
         totalSupply -= aux2;
      }
   }
   aux = penaltyCount[_user]; //get the penalty count from the user
   penaltyCount[_user] = aux + negativePoints; //update the penalty number +5
   // 25 penalty point equals facount frozen
   if (penaltyCount[_user] >= 25){
      frozenAccount[_user] = true;
      FrozenFunds(_user, true);
```

```
        }
      }
    }

    //*****************************
    //******ADD USER FUNCITON ******
    function addUser(address publicKey, string id) onlyOwner {
      findUserStep1[count] = publicKey;
      findUserStep2[publicKey] = id;
      penaltyCount[publicKey] = 0; //initialize penalty in 0
      count++;
    }
    function addUserEC(uint idEC, string id) onlyOwner {
      findUserEC[idEC] = id;
      countEC++;
    }
}
```

9.5. Digital Support

In order to make the project more understandable, two videos have been recorded.

Smart Token for the campus using blockchain

This video serves as a project introduction while showing how the technology behind the project works. It can be found on the following link:

https://www.youtube.com/watch?v=5m7bXAMPSrU&t=6s

NUST Wallet Demo

The following video introduces the interface created for the students and shows all its possibilities. It can be found on the following link:

https://www.youtube.com/watch?v=DQSgAcTPHhI&feature=youtu.be

9.6. Smart Contract: Upload Files and Images

```
pragma solidity 0.4.11;

contract uploadFile {

  uint256 public fileCount;

   /* This creates an array with all balances */

  mapping (uint256 => address) public getAuthor;
  mapping (uint256 => string) public getData;
```

```
    function pushByte(string b) {

    getAuthor[fileCount]=msg.sender;
    getData[fileCount] = b;

    fileCount++;
  }

  /* This unnamed function is called whenever someone tries to send ether to it */

  function () {

    throw;     // Prevents accidental sending of ether
  }
}
```

10. BIBLIOGRAPHY

Bitcoin.org. (2017). Developer Guide - Bitcoin. [online] Available at: https://bitcoin.org/en/developer-guide#block-chain [Accessed 21 Mar. 2017].

Ethereum.org. (2017). Ethereum Project. [online] Available at: https://www.ethereum.org/ [Accessed 15 Mar. 2017].

En.bitcoin.it. (2017). Full node - Bitcoin Wiki. [online] Available at: https://en.bitcoin.it/wiki/Full_node [Accessed 23 Mar. 2017].

Solidity.readthedocs.io. (2017). Introduction to Smart Contracts — Solidity 0.4.12 documentation. [online] Available at: http://solidity.readthedocs.io/en/develop/introduction-to-smart-contracts.html [Accessed 23 Mar. 2017].

Lombardo, H. (2017). Blockchain Serves as Tool for Human, Product and IoT Device Identity Validation. [online] Chain of Things. Available at: https://www.chainofthings.com/news/2017/1/11/blockchain-serves-as-tool-for-human-product-and-iot-device-identity-validation [Accessed 23 Mar. 2017].

Bankingtech.com. (2017). Standard Chartered explores blockchain viability » Banking Technology. [online] Available at: http://www.bankingtech.com/514402/standard-chartered-explores-blockchain-viability/ [Accessed 5 Apr. 2017].

Stefan, Stefan and Stefan (2017). Using Blockchain to Secure IoT | Dyogram. [online] Dyogram. Available at: http://www.dyogram.com/2016/08/using-blockchain-to-secure-iot/ [Accessed 5 Apr. 2017].

Ethereum.github.io. (2017). [online] Available at: Ethereum.github.io. (2017). Go Ethereum Downloads. [online] Available at: https://ethereum.github.io/go-ethereum/downloads/ [Accessed 23 Mar. 2017]. [Accessed 5 Apr. 2017].

Chris Dannen . *Introducing Ethereum and Solidity. Foundations of Cryptocurrency and Blockchain Programming for Beginners*, 2017.

Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller and Steven Goldfeder. *Bitcoin and Cryptocurrency Technologies*, Chapter 1 to 7, 2015.

Jacob Stenum, Nikolaj Zangenberg and Simon Oliver. The use of block chain technology in different application domains, The IT University of Copenaghen 20th, May 2015.